

# 시뮬레이터 기본교육 및 Control & Planning 알고리즘 개발

프로젝트 지향 자율주행차 전문인력 양성과정

# 목차

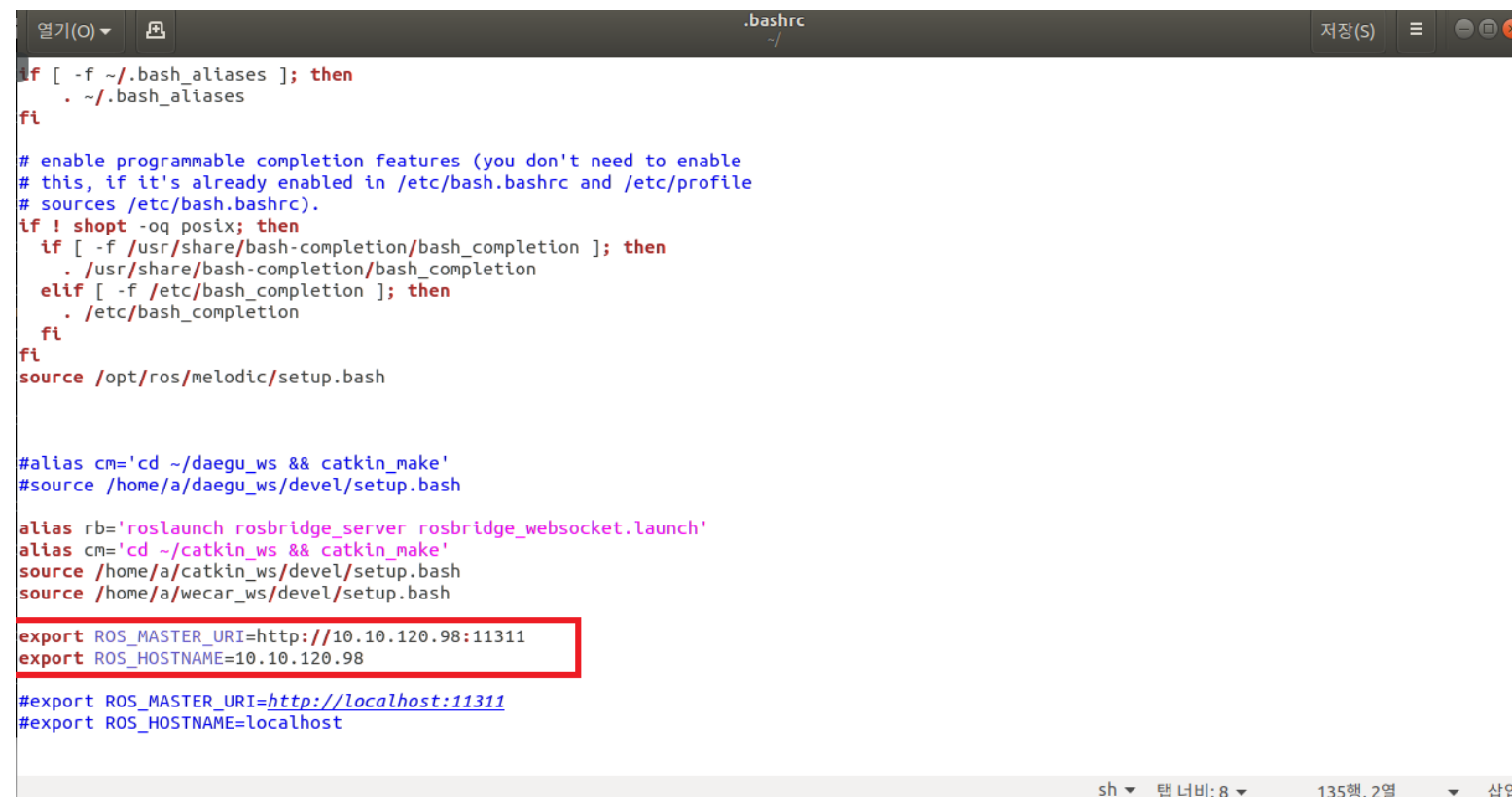
---

1. Simulator - WeCar 연동
2. 라이다 데이터 받기
3. 좌표 변환
4. 예제

# **1. Simulator - WeCar 연동**

# Simulator - WeCar 연동

- Simulator PC 세팅
  - wecar와 같은 로컬 네트워크로 연결해줌.(wifi, 유선랜)
  - IP 확인하기
    - \$ ifconfig
  - ROS를 Master 잡기 위해 bashrc를 수정 한다
    - \$ gedit ~/.bashrc
    - 수정 후 source 명령어를 이용해 변경 내용 적용
    - \$ source ~/.bashrc
  - ROS Bridge를 실행해서 시뮬레이터와 연동
    - \$ roslaunch rosbridge\_server rosbridge\_websocket.launch



```
if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
    if [ -f /usr/share/bash-completion/bash_completion ]; then
        . /usr/share/bash-completion/bash_completion
    elif [ -f /etc/bash_completion ]; then
        . /etc/bash_completion
    fi
fi
source /opt/ros/melodic/setup.bash

#alias cm='cd ~/daegu_ws && catkin_make'
#source /home/a/daegu_ws/devel/setup.bash

alias rb='roslaunch rosbridge_server rosbridge_websocket.launch'
alias cm='cd ~/catkin_ws && catkin_make'
source /home/a/catkin_ws/devel/setup.bash
source /home/a/wecar_ws/devel/setup.bash

export ROS_MASTER_URI=http://10.10.120.98:11311
export ROS_HOSTNAME=10.10.120.98

#export ROS_MASTER_URI=http://localhost:11311
#export ROS_HOSTNAME=localhost
```

# Simulator - WeCar 연동

- Wecar 세팅
  - ip확인
  - master는 simulator pc로하고, host는 wecar ip로 bashrc를 수정 후 적용하기
  - Vesc 드라이버를 실행
    - \$ roslaunch vesc\_driver vesc\_node.launch

```
nvidia@tegra-ubuntu: ~  
usb0    Link encap:Ethernet  HWaddr 52:37:38:0e:e4:bd  
UP BROADCAST MULTICAST  MTU:1500  Metric:1  
RX packets:0 errors:0 dropped:0 overruns:0 frame:0  
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0  
collisions:0 txqueuelen:1000  
RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)  
  
usb1    Link encap:Ethernet  HWaddr 6a:c8:f4:b8:fe:ab  
UP BROADCAST MULTICAST  MTU:1500  Metric:1  
RX packets:0 errors:0 dropped:0 overruns:0 frame:0  
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0  
collisions:0 txqueuelen:1000  
RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)  
  
wlan0   Link encap:Ethernet  HWaddr 00:04:4b:f6:d4:b0  
inet addr:10.10.120.56  Bcast:10.10.121.255  Mask:255.255.254.0  
inet6 addr: fe80::305c:8a09:2c3c:9d4c/64 Scope:Link  
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1  
RX packets:485649 errors:0 dropped:0 overruns:0 frame:0  
TX packets:113494 errors:0 dropped:0 overruns:0 carrier:0  
collisions:0 txqueuelen:1000  
RX bytes:365741130 (365.7 MB)  TX bytes:8660462 (8.6 MB)  
  
nvidia@tegra-ubuntu:~$
```

```
.bashrc (~/) - gedit  
# colored GCC warnings and errors  
#export GCC_COLORS='error=01;31:warning=01;35:note=01;36:caret=01;32:locus=01:quote=01'  
  
# some more ls aliases  
alias ll='ls -alF'  
alias la='ls -A'  
alias l='ls -CF'  
  
# Add an "alert" alias for long running commands.  Use like so:  
# sleep 10; alert  
alias alert='notify-send --urgency=low -i "${?} = 0" && echo terminal || echo error"  
"${history|tail -n1|sed -e '\''s/^s*[0-9]*\s*//;s/[:&]\s*alert$//'\''}'"  
  
# Alias definitions.  
# You may want to put all your additions into a separate file like  
# ~/.bash_aliases, instead of adding them here directly.  
# See /usr/share/doc/bash-doc/examples in the bash-doc package.  
  
if [ -f ~/.bash_aliases ]; then  
    . ~/.bash_aliases  
fi  
  
# enable programmable completion features (you don't need to enable  
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile  
# sources /etc/bash.bashrc).  
if ! shopt -oq posix; then  
    if [ -f /usr/share/bash-completion/bash_completion ]; then  
        . /usr/share/bash-completion/bash_completion  
    elif [ -f /etc/bash_completion ]; then  
        . /etc/bash_completion  
    fi  
fi  
export PATH=/usr/local/cuda-9.0/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin  
export LD_LIBRARY_PATH=/usr/local/cuda-9.0/lib64:  
source /opt/ros/kinetic/setup.bash  
source ~/wecar_ws/devel/setup.bash  
  
export ROS_MASTER_URI=http://10.10.120.98:11311  
export ROS_NAME=10.10.120.56  
export ROS_IP=10.10.120.56  
export LD_LIBRARY_PATH=/usr/local/lib:/opt/ros/kinetic/lib:/opt/ros/kinetic/lib/aarch64-linux-gnu:/usr/local/cuda-9.0/lib64  
sh Tab Width: 8 Ln 124, Col 29 INS
```

# Simulator - WeCar 연동

---

- Wecar 세팅
  - 제어 명령 보내기
  - `$ rostopic pub /commands/motor/speed std_msgs/Float64 "data: 2000.0" -r 10`
  - `$ rostopic pub /commands/servo/position std_msgs/Float64 "data: 0.15" -r 10`

`rpm = velocity(m/s) * rpm_gain`

`servo value = steering_angle_to_servo_gain * steering angle (radians) + steering_angle_to_servo_offset`

`rpm_gain = 4614`

`steering_angle_to_servo_gain = -1.2135`

`steering_angle_to_servo_offset: 0.5304`

`servo value range : 0.15(LEFT) ~ 0.5304(MID) ~ 0.85(RIGHT)`

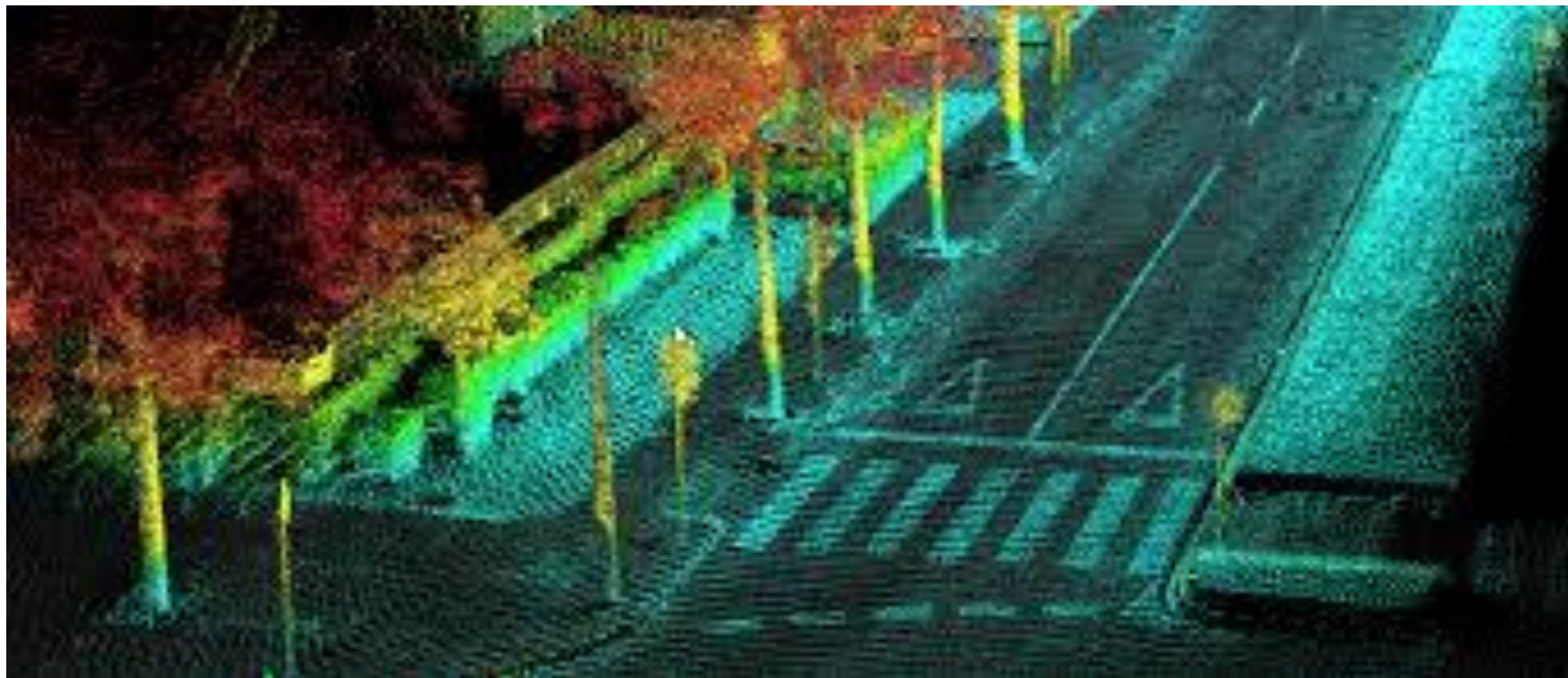
## **2. 라이다 센서 데이터 받기**

# 라이다 데이터 받기

---

- 라이다

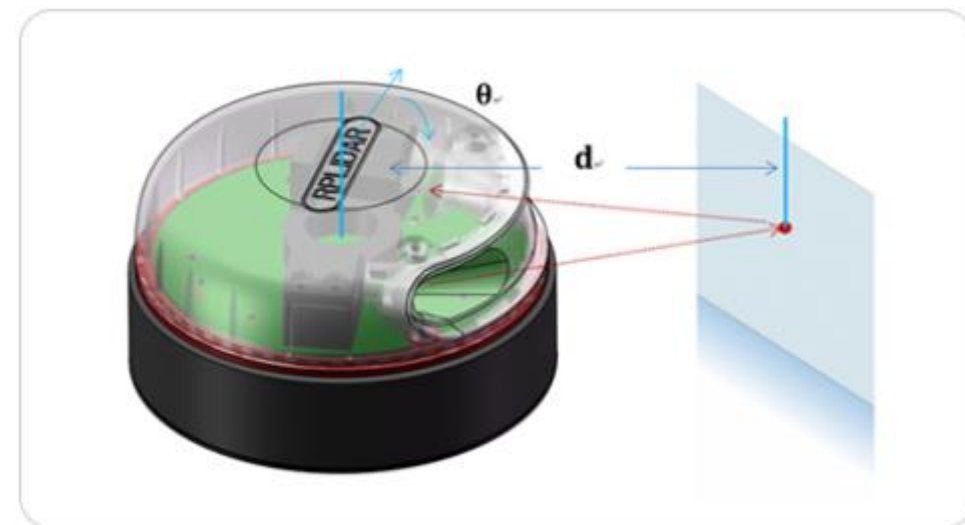
- Lidar(Light Detection And Ranging)는 레이저 광펄스를 여러 범위로 쏘아 돌아오는 시간을 계산해 물체와의 거리를 측정할 수 있다.
- 사물의 형태 인식이 가능하고 정밀도가 높다
- (레이더와 비교시)탐지 거리가 비교적 짧고, 기상 상황이나 주행 환경에 민감하다.






# 라이다 데이터 받기

- RPLIDAR A2
  - Wecar에서 사용하는 라이다 모델
  - ROS 패키지 사용(<http://wiki.ros.org/rplidar>)



 <b>12</b> meters	 <b>8000</b> Sa/s	 <b>1</b> <sup>*</sup> Degree	 <b>0.2</b> <sup>**</sup> cm	 <b>5</b> Volt
Detection Range	Sample Rate	Angular Res.	Distance Res.	Power Supply

# 라이다 데이터 받기

---

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import rospy
from sensor_msgs.msg import LaserScan, PointCloud
from std_msgs.msg import Float64
from vesc_msgs.msg import VescStateStamped
from math import cos, sin, pi
from geometry_msgs.msg import Point32

class simple_controller :

    def __init__(self):
        rospy.init_node('simple_controller', anonymous=True)
        rospy.Subscriber("/scan", LaserScan, self.laser_callback)

        while not rospy.is_shutdown():
            rospy.spin()

    def laser_callback(self, msg):

        for theta, r in enumerate(msg.ranges) :
            print(theta, r)

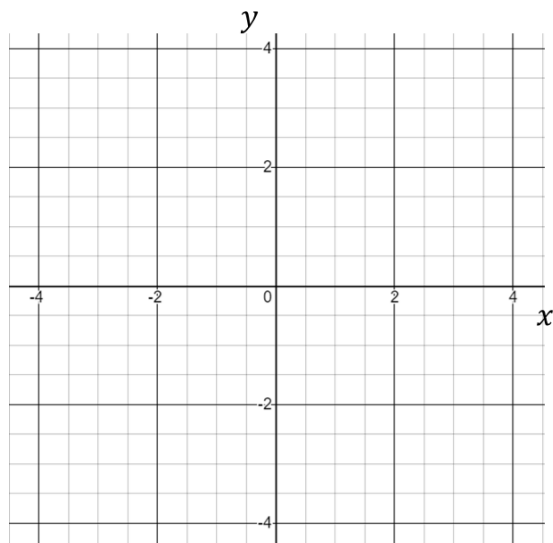
if __name__ == '__main__':
    try:
        test_track = simple_controller()
    except rospy.ROSInterruptException:
        pass
```

### 3. 좌표변환

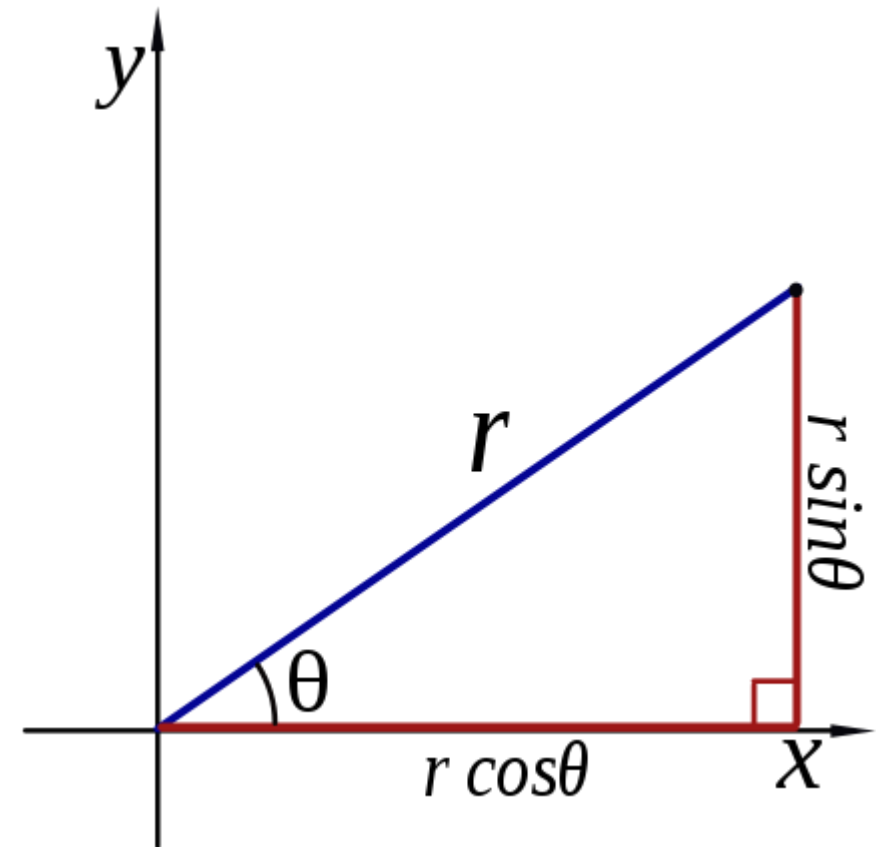
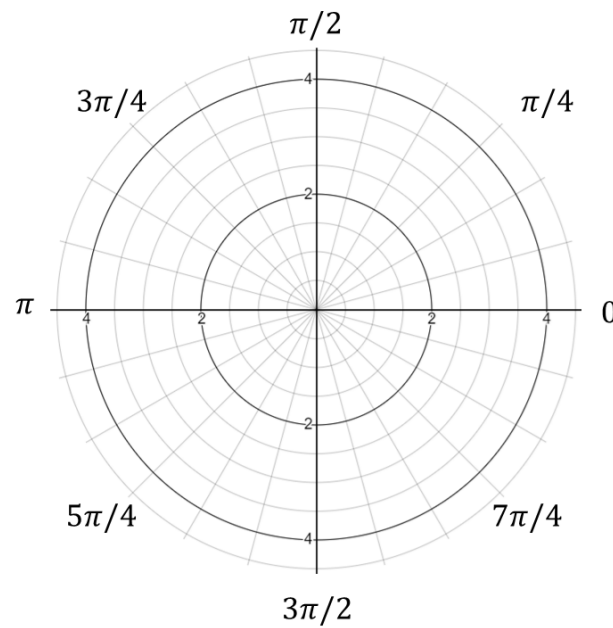
# 좌표변환

- 극좌표계 -> 직교좌표계
  - 필요에 따라 변환해서 사용

직교좌표계



극좌표계



# 좌표변환

---

```
class simple_controller :

    def __init__(self):
        rospy.init_node('simple_controller', anonymous=True)
        rospy.Subscriber("/scan", LaserScan, self.laser_callback)

        self.pcd_pub = rospy.Publisher('laser2pcd',PointCloud, queue_size=1)

        while not rospy.is_shutdown():
            rospy.spin()

    def laser_callback(self,msg):
        pcd=PointCloud()
        motor_msg=Float64()
        pcd.header.frame_id=msg.header.frame_id
        angle=0

        for r in msg.ranges :

            tmp_point=Point32()
            tmp_point.x=r*cos(angle)
            tmp_point.y=r*sin(angle)
            print(angle,tmp_point.x,tmp_point.y)
            angle=angle+(1.0/180*pi)
            if r<12 :
                pcd.points.append(tmp_point)

        self.pcd_pub.publish(pcd)

if __name__ == '__main__':
    try:
        test_track=simple_controller()
    except rospy.ROSInterruptException:
        pass
```

## 4. 예제

# 예제

---

- 전방에 장애물이 있으면 정지하기

```
class simple_controller :
    def __init__(self):
        rospy.init_node('simple_controller', anonymous=True)
        rospy.Subscriber("/scan", LaserScan, self.laser_callback)
        self.motor_pub = rospy.Publisher('commands/motor/speed', Float64, queue_size=1)
        self.servo_pub = rospy.Publisher('commands/servo/position', Float64, queue_size=1)
        self.pcd_pub = rospy.Publisher('laser2pcd', PointCloud, queue_size=1)

        while not rospy.is_shutdown():
            rospy.spin()

    def laser_callback(self, msg):
        pcd=PointCloud()
        motor_msg=Float64()
        pcd.header.frame_id=msg.header.frame_id
        angle=0
        for r in msg.ranges :
            tmp_point=Point32()
            tmp_point.x=r*cos(angle)
            tmp_point.y=r*sin(angle)
            angle=angle+(1.0/180*pi)
            if r<12 :
                pcd.points.append(tmp_point)

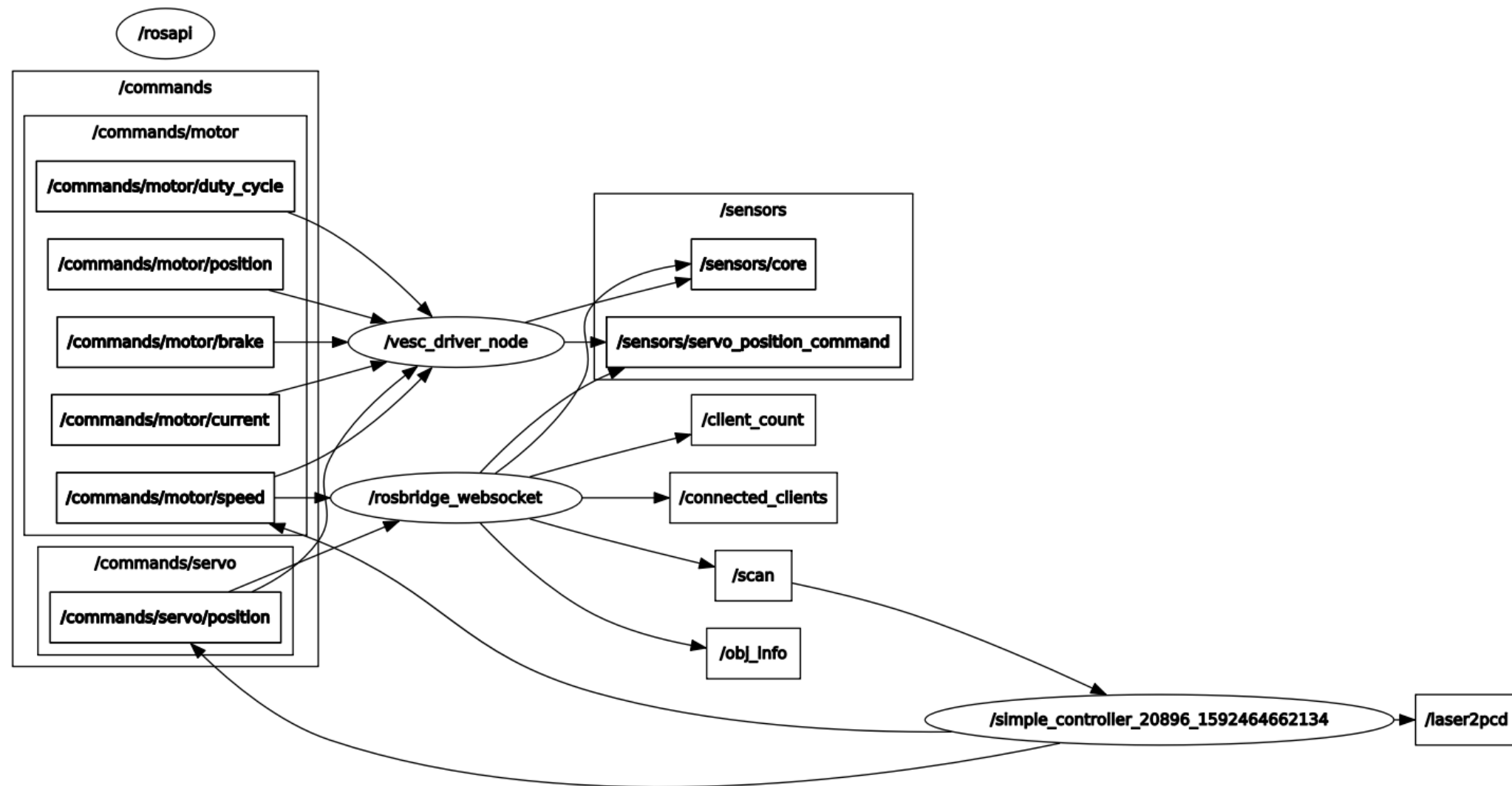
        count=0
        for point in pcd.points :
            if point.x > 0 and point.x <1 :
                count=count+1

        if count > 20:
            motor_msg.data=0
        else :
            motor_msg.data=2000

        print(count)
        self.motor_pub.publish(motor_msg)
        self.pcd_pub.publish(pcd)
```

# 예제

- 예제에서 사용한 코드를 Wecar와 연동해서 확인하기
  - rqt\_graph 모습





**END**